Piecewise-Linear Manifold Learning: A Heuristic Approach to Nonlinear Dimensionality Reduction

Strange, Harry; Zwiggelaar, Reyer

Published in:
Intelligent Data Analysis

DOI:
10.3233/IDA-150779

Publication date:
2015

Citation for published version (APA):
Strange, H., & Zwiggelaar, R. (2015). Piecewise-Linear Manifold Learning: A Heuristic Approach to Nonlinear Dimensionality Reduction. *Intelligent Data Analysis*, *19*(6), 1213-1232. https://doi.org/10.3233/IDA-150779

Piecewise-Linear Manifold Learning: A Heuristic Approach to Non-linear Dimensionality Reduction

Harry Strange* and Reyer Zwiggelaar

Department of Computer Science,
Aberystwyth University,
Aberystwyth,
Wales,
SY23 3DB UK.

December 31, 2014

Abstract

The need to reduce the dimensionality of a dataset whilst retaining its inherent manifold structure is key to many pattern recognition, machine learning, and computer vision problems. This process is often referred to as manifold learning since the structure is preserved during dimensionality reduction by learning the intrinsic low-dimensional manifold that the data lies upon. In this paper a heuristic approach is presented to tackle this problem by approximating the manifold as a set of piecewise linear models. By merging these linear models in an order defined by their global topology a globally stable and locally accurate model of the manifold can be obtained. A detailed analysis of the proposed approach is presented along with comparison with existing manifold learning techniques. Results obtained on both artificial and image based data show that in many cases this heuristic approach to manifold learning is able to out-perform traditional techniques.

1 Introduction

The field of manifold learning stems from the need to make sense of high-dimensional data, that is, datasets with large numbers of samples and variables. Advances in data collection

*Electronic address: hgs08@aber.ac.uk; Corresponding author

devices has meant that in many cases, more variables are measured than are actually needed. This redundancy can mean that the important variables, or dimensions, are 'hidden' within the dataset. Traditional statistical analysis tools either fail to handle datasets with large numbers of dimensions or significantly degrade in performance. As such, it is useful to have a tool that can extract these hidden dimensions and remove the redundant ones. This reduction of the number of dimensions can also help with visualising the data, which enables researchers to observe the data within a known frame of reference. This visualisation process can make it easier for conclusions to be drawn as to the structure and properties of the high-dimensional data. Manifold learning is one such tool that can reduce the dimensionality of the data whilst retaining important information contained within the data.

At the core of manifold learning is the assumption that the high-dimensional data lies on, or near, a low-dimensional manifold. In recent years this assumption has been shown to hold both experimentally, through the use of manifold learning on data and visualising the results (e.g. [28, 33, 25]), and, in part, theoretically [23]. The heart of this argument is that manifolds are fundamental to perception, so the human brain must have some way of representing them. Due to the fact that the possible images of an object lie on a manifold, it has been hypothesised that a visual memory is stored as a manifold of stable states or a continuous attractor [23].

In this paper a new manifold learning algorithm is presented: Piecewise-Linear Manifold Learning (PLML). This algorithm differs from most other approaches to manifold learning as it is based on a simple heuristic: the global shape of the manifold can be built by merging neighbouring local models. The PLML algorithm builds on the strengths of linear techniques such as Principal Components Analysis [11], and also recent advances in global alignment of local linear models (e.g. Local Tangent Space Analysis [35], Local Linear Coordination [27], and Manifold Charting [3]). The core of the PLML algorithm is the merging of local models according to their topology as defined by a Minimum Spanning Tree. Unlike existing approaches where a global alignment is attained via optimisation of a specific cost function, the PLML approach finds the low-dimensional embedding through the iterative alignment of local models.

The remainder of this paper is structured as follows. Section 2 provides an overview of existing manifold learning algorithms and states the contribution of the proposed algorithm to the field. In Section 3 the Piecewise-Linear Manifold Learning algorithm is described in a step

by step manner, and Section 4 provides analysis of its performance. Section 5 shows the results of the proposed approach, compared against leading manifold learning algorithms, on artificial and image data. Finally, conclusions to this work are drawn in Section 6.

2 Existing Work

Manifold learning has become an established field of research with much attention being given to algorithms that achieve robust embeddings of highly non-linear data. Most, if not all, will have their origins in Principal Components Analysis (PCA) [11]. PCA is a linear approach to the manifold learning problem which projects the data onto the hyperplane of maximum variance. This maximum variance hyperplane is found by measuring the covariance between all of the data points and then projecting the original data points onto the eigenvectors of this covariance matrix. As such, PCA is only capable of finding a global linear hyperplane within the data and will not find meaningful embeddings for highly non-linear datasets. Similar to PCA is Multidimensional Scaling (MDS) [6] which replaces the covariance matrix with an inter point Euclidean distance matrix. However, since MDS only measures the linear distance relations between points it is still unable to recover non-linear structure from the data. To overcome this linear constraint many non-linear, or manifold learning, algorithms have been presented that aim to maintain certain properties of the data between the high and low-dimensional spaces [1, 3, 12, 20, 28, 21, 27, 33, 35].

Manifold learning algorithms can be broadly split into three classes based on the properties of the data that they retain: global; local; or both local and global. Isomap [28] is perhaps the most widely known, and widely used, global manifold learning algorithm. It builds a low-dimensional representation of the high-dimensional data by retaining the global geodesic distances between points. As such, it can be seen to be a non-linear extension of MDS with the Euclidean distance matrix being replaced by a geodesic distance matrix. Other global algorithms include Kernel PCA [21], where the manifold learning problem is tackled using a kernel phrasing of classic PCA, and Diffusion Maps [12], which is grounded in the field of dynamical systems and used the diffusion distance to measure global properties of the manifold.

Local manifold learning algorithms aim to build a low-dimensional embedding by retaining

the local structure of the data. Arguably the most widely known local algorithm is Locally Linear Embedding (LLE) [20]. LLE aims to reconstruct local neighbourhoods in the low-dimensional space via a set of locally linear fits around each data point. These 'fits' are found in the high-dimensional space and then reconstructed in the low-dimensional space by a least squares method. Other notable local manifold learning algorithms include Laplacian Eigenmaps [1], where the graph Laplacian is used to reconstruct the manifold, and Maximum Variance Unfolding [33] where semidefinite programming is used to 'unfold' the neighbourhood graph.

The third class of manifold learning algorithms, local/global, aim to reconstruct the local properties of the manifold over a global scale. That is, local models are built across the manifold and then a global alignment of these models is approximated. As such, this class of manifold learning algorithm attempts to draw together the strengths of both global and local manifold learning algorithms. Local Tangent Space Alignment (LTSA) [35] models the local structure of the data using local tangent spaces about each datapoint which are then globally aligned to produce the low-dimensional embedding. Core to the LTSA algorithm is the observation that, if local linearity is assumed, then there exists a linear mapping from a datapoint to its local tangent space in the high-dimensional space. Conversely, there exists a linear mapping from the corresponding low-dimensional datapoint to the same local tangent space [35]. As such, LTSA searches for the coordinates of the low-dimensional data at the same time as it searches for the linear mappings of the low-dimensional data to the tangent spaces of the high-dimensional data.

Manifold Charting [3] and Local Linear Coordination (LLC) [27] are two methods that seek to obtain a low-dimensional embedding by aligning a set of local models. Both methods start by forming a mixture of local linear models using either a Mixture of Factor Analysers or a Mixture of Probabilistic PCA models[1]. LLC proceeds to align these local linear models by minimising a variant of the LLE [20] cost function. The cost function employed by Manifold Charting is based on minimising the 'disagreement' between the linear models on the global coordinates of the data points. Both techniques employ an eigensolver approach to form the low-dimensional alignment. Although these local/global algorithms attempt to tie together the strengths of both local and global manifold learning algorithms, they are not without their weaknesses. The quality of the low-dimensional embedding is highly dependent on the correct choice of parameters, as such

[1]Either method could be used, however traditionally both use a Mixture of Factor Analysers approach

multiple runs of the algorithms with different combinations of parameters are often needed to find the optimal embedding. Also, as pointed out in [30], the fitting of the mixture of factor analysers is susceptible to the presence of local maxima in the log-likelihood function.

The developed method, Piecewise-Linear Manifold Learning (PLML), fits into this local/global category of manifold learning algorithms as it builds a set of local models that are then globally aligned to form the global low-dimensional representation. However, it does not require the minimisation of a cost function or the solution to a large eigenproblem. Rather, it builds the low-dimensional alignment in a piecewise manner by merging consecutive local models. As such, PLML is efficient, easy to understand, and easy to implement. In addition, as is shown in Sections 4 and 5, this heuristic methodology out-performs existing approaches on a range of artificial and real world examples.

3 Piecewise-Linear Manifold Learning

The Piecewise-Linear Manifold Learning (PLML) algorithm can be split into four main steps. First, the data is partitioned into local PCA models via k-means clustering. Then, in the second step the connectivity of the local models is determined by forming a Minimum Spanning Tree using the cluster centres as nodes. The third step builds a global model of the data by traversing the Minimum Spanning Tree and merging the models during traversal. Finally, the low-dimensional representation is found by performing PCA on the global model found in the third step. A diagram showing how these steps are related is shown in Fig. 1.

The PLML algorithm takes as input a high-dimensional set of n samples $\mathbf{X} = \{\mathbf{x}_i\}_{i=1}^n \in \mathbb{R}^p$ sampled from a low-dimensional manifold $\mathcal{M} \in \mathbb{R}^q$ embedded within $\mathbb{R}^p$ (where $q \ll p$). Each sample, $\mathbf{x}_i$, is represented as a row vector within the input matrix $\mathbf{X}$. The goal is to recover a set of samples $\mathbf{Y} = \{\mathbf{y}_i\}_{i=1}^n \in \mathbb{R}^q$ that best approximate the q-dimensional manifold $\mathcal{M}$. At a local scale it is assumed that the manifold $\mathcal{M}$ is homeomorphic to Euclidean space and is a C^∞-manifold (i.e. it is smooth differentiable). The two free parameters are the target dimensionality, q, and neighbourhood size parameter, k, which is used to determine the size of the clusters formed in the first step.

The initial step in the PLML algorithm it to create local models by partitioning $\mathbf{X}$ into

$c = \lfloor n/k \rfloor$ clusters with the constraint that no cluster is larger than the pre-defined value h. This ensures that, as far as possible, all clusters are created with a similar number of samples. In practice the value of h is set to k since clusters need to be created with as near to equal size as possible. A constrained clustering algorithm is used to partition $\mathbf{X}$ into c clusters $\{C_l\}_{l=1}^{c}$ such that the distance between each sample, $\mathbf{x}_i$, and its nearest cluster centre, $\bar{C}_l$ is minimised:

$$\min_{C_1,\ldots,C_c} \sum_{i=1}^{n} \min_{l=1,\ldots,c} \left(\frac{1}{2} \parallel \mathbf{x}_i - \bar{C}_l \parallel^2 \right) \tag{1}$$

with the specific constraint that no cluster, C_l, is smaller than the minimum cluster size, h. The above approach is similar to that proposed in [2], where a normal k-means clustering algorithm is run with the specific cluster size constraint added.

Once $\mathbf{X}$ has been partitioned into c clusters, local linear models can be formed. A local model relating to the l-th cluster is defined as the points in C_l projected onto the hyperplane spanned by the basis vectors $\mathbf{U}_l$. The basis vectors, $\mathbf{U}_l$, are found by performing local PCA on the points in $\mathbf{X}_{C_l}$. The points are then projected onto the subspace spanned by the top q vectors in $\mathbf{U}_l$ such that

$$\mathbf{x'}_i = \mathbf{x}_i \mathbf{U}_l \mathbf{U}_l^T + (\bar{C}_l - (\bar{C}_l \mathbf{U}_l \mathbf{U}_l^T)) \tag{2}$$

where $\mathbf{x}_i \in C_l$, $\mathbf{x}_i \mathbf{U}_l \mathbf{U}_l^T$ projects $\mathbf{x}_i$ onto the subspace spanned by the column vectors given in $\mathbf{U}$ and $(\bar{C}_l - (\bar{C}_l \mathbf{U}_l \mathbf{U}_l^T))$ is the translation vector that moves $\mathbf{x}_i \mathbf{U}_l \mathbf{U}_l^T$ onto the local model. Here, $\mathbf{U}_l$ represents the top q basis vectors ordered according to the eigenvalues of the local PCA models. This ensures that the points within a local model are locally q-dimensional even though the model is embedded within p-dimensional space. The local model containing the data points $\mathbf{X}_{C_l}$ projected onto their local subspace is referred to as Π_l, such that, $\Pi_l = \{\mathbf{x'}_i\}_{i=1}^{|C_l|} \; \forall \; \mathbf{x}_i \in C_l$.

Since at this stage the data is locally q-dimensional the next step is to align the local models to build a global alignment of the local models. This global alignment will still be embedded in p-dimensional space but will be locally and globally q-dimensional, so a simple PCA transformation will reduce the dimensionality of the data to q-dimensions.

This global alignment step is performed by iteratively aligning the basis vectors of neighbouring models and projecting previously visited points onto the current model's basis vectors

as shown in Fig. 2. The global model is iteratively formed by merging visited models into a global representation of the data. A Minimum Spanning Tree (MST) provides a representation of the topology of the local models and so forms the foundation for the order in which the models are visited for alignment. A definition of the model level MST is given by

$$G(i,j) = \begin{cases} d(\bar{C}_i, \bar{C}_j) & \text{if} \quad \exists e \in E \mid e = \langle v_a, v_b \rangle \\ & \qquad v_a \in C_i, v_b \in C_j \\ \infty^+ & \text{otherwise} \end{cases} \tag{3}$$

where $d(\bar{C}_i, \bar{C}_j)$ is the Euclidean distance between the cluster centre, $\bar{C}_i$, of C_i and the cluster centre, $\bar{C}_j$, of C_j. Since the graph produced using the above process is not always ensured to be a MST, the MST algorithm is run on the resulting graph G to ensure that an MST is produced. To walk along the MST a simple pre-order traversal [29] is used as this ensures that parents are visited before children, and siblings are visited in left-to-right order.

The alignment and projection of points can either occur as *forward projection*, or as *backward projection*. Forward projection occurs when the graph traversal moves onto a new node that has not been previously visited. Conversely, backward projection occurs when the graph traversal is re-visiting a previously visited node. Each projection needs to be treated slightly differently to avoid discontinuities in the overall projection process. The need for a separate backward projection step becomes apparent when you consider that the hyperplanes spanned by two models are too far apart to perform forward projection. Therefore, the model is aligned to the previously visited nodes' actual representation to ensure that no distortions occur as a result of backward projection (Fig. 2).

For forward projection the previously visited models are projected onto the current model so that they lie in the same co-ordinate space. The previously visited models all lie on the same co-ordinate space and they need to be projected onto the co-ordinate space of the current model. However, this projection process is more involved than a simple orthogonal projection (such as that described in Equation 2) as the projection uses basis vectors that are not in the same locally linear model. As such this can introduce distortions into the mappings. To reduce distortions the two models are required to be parallel and it is this parallelisation of projections that is one of the key concepts within the PLML algorithm.

The principal insight is that two models are parallel if their normal vectors are aligned. This is equivalent to the dihedral angle between the two models being equal to 0. Therefore, the goal becomes finding the rotation needed to align the previous model with the current model such that they are parallel. Given the basis vectors of two models $\mathbf{U}_i$ and $\mathbf{U}_j$, the goal is to find the rotation matrix needed to align $\mathbf{U}_i$ to $\mathbf{U}_j$ and then apply this rotation to the points in Π_i prior to projection onto the subspace defined by $\mathbf{U}_j$. Throughout, the $(q+1)$ basis vectors are used with the $(q+1)$-th basis vector providing a normal vector to the hyperplane. Therefore, when referred to the matrix $\mathbf{U}_i$, the matrix is of size $p \times (q+1)$.

Finding the rotation matrix can be easily phrased as a point cloud registration problem. The two sets of basis vectors, one for the previous model and one for the current model, are transformed to point clouds by applying a unique scaling factor to each axis using the $p \times (2q+2)$ matrix:

$$\mathbf{S}(i,j) = \begin{cases} j & \text{if}(j \leq (q+1)) \\ j + (\frac{1}{2}) & \text{otherwise} \end{cases} \tag{4}$$

such that the scaled basis vectors become $\mathbf{M}_i = [\mathbf{U}_i \ {}^{-}\mathbf{U}_i] \circ \mathbf{S}$. The matrix, ${}^{-}\mathbf{U}_i$ is the additive inverse of $\mathbf{U}_i$ which mirrors the axes so that alignments do not occur whereby two axes are aligned in opposite directions. The scaling ensures that there is a unique solution to the registration. Given the two axis matrices $\mathbf{M}_i$ and $\mathbf{M}_j$ the rotation matrix $\mathbf{R}$ is found so that $\| \mathbf{M}_i - \mathbf{M}_j\mathbf{R} \|$ is minimised (where $\| \cdot \|$ is the Frobenius norm). The matrix $\mathbf{R}$ is orthonormal and gives the rotation needed to align the axis matrix $\mathbf{M}_i$ to $\mathbf{M}_j$. Thus, orthogonal Procrustes [22] can be used to obtain the optimal solution via the Singular Value Decomposition (SVD) of $\mathbf{M}_i^T\mathbf{M}_j$. That is, if the SVD of $\mathbf{M}_i^T\mathbf{M}_j$ is $\mathbf{U}\Sigma\mathbf{V}^T$ then $\mathbf{R} = \mathbf{U}\mathbf{V}^T$.

The rotation matrix is then applied by $\Pi_i' = \Pi_i\mathbf{R}$. Now that the two models are parallel, Π_i' is projected onto the subspace spanned by the points in Π_j as follows:

$$\Pi_i' = \Pi_i'\mathbf{U}_j\mathbf{U}_j^T + (\bar{C}_j - (\bar{C}_j\mathbf{U}_j\mathbf{U}_j^T)) \ \forall \ i \in \Psi \tag{5}$$

where Ψ is a list containing the indices of previously visited models.

If the next model in the traversal has already been visited then the parallel projections

process described above does not need to take place. Rather, the model is aligned with its original representation (Fig. 3). This is because the global model (that is the hyperplane of all previously visited models) may be too far away from the current model to perform forward projection without distortion. Therefore, the fact that the current model is represented within the global model is exploited so that, if the two are aligned then the global model's hyperplane is correctly aligned with the current model's hyperplane. Consider the case of aligning Π_i' to Π_i, the translation vector that moves the mean point of Π_i' onto the mean point of Π_i and also the rotation matrix that correctly aligns the samples in Π_i' to Π_i need to be found. This step can easily be achieved by running Procrustes analysis on the two sets of samples within the two models. Since the number of samples within the models will not have changed and the forward projection step described above does not distort the samples within the model, Procrustes analysis [9] can be used to find the rotation matrix $\mathbf{R}$ to match Π_i' to Π_i. The translation v and rotation matrix $\mathbf{R}$ can be found as follows. The translation vector is simply defined as $v = \bar{\Pi}_i - \bar{\Pi}_i'$. For the rotation matrix let $\mathbf{Z} = \Pi_i^T \Pi_j$ and its Singular Value Decomposition (SVD) as $\mathbf{Z} = \mathbf{ULV}^T$ [16]. The Procrustes rotation matrix is given by $\mathbf{R} = \mathbf{UV}^T$ and the translation vector is given by $v = \bar{\Pi}_i - \mathbf{R}\bar{\Pi}_i'$ (where $\bar{\Pi}$ represents the mean point of the model) [24]. Therefore, when performing Backward alignment, $\Pi_i' = \Pi_i'\mathbf{R} + v$. As with the forward projection step this translation and rotation is applied to all previously visited models rather than just the single previously visited model.

Once all nodes have been visited in the traversal described above, a globally aligned q-dimensional representation of the manifold embedded within p-dimensional space will have been obtained. This representation is contained in the models $\{\Pi_l'\}_{l=1}^c$. The matrix $\mathbf{Q} = \{\Pi_l'\}_{l=1}^c$ is set to contain all the samples of the globally aligned representation. To find the low-dimensional embedding PCA is performed on $\mathbf{Q}$ such that

$$\Lambda \mathbf{V} = \mathbf{C_Q V} \tag{6}$$

where $\mathbf{C_Q}$ is the covariance matrix of $\mathbf{Q}$ and $\mathbf{V}$ is a matrix containing as columns the top q-dimensional eigenvectors sorted according to their associated eigenvalues, Λ. The low-dimensional representation $\mathbf{Y}$ is then given by $\mathbf{Y} = \mathbf{QV}$.

4 Analysis

Due to the heuristic nature of the PLML algorithm, an in-depth analysis is needed to enable a better understanding as to how the algorithm performs under different data conditions. This section explores the performance of the PLML algorithm under different data densities and parameter values. The performance of the algorithm is measured using three different quality assessment measures and the algorithm is compared against four leading manifold learning algorithms. These algorithms are chosen so as to represent each of the different approaches to manifold learning. Principal Components Analysis (PCA) [11] is used to indicate the performance of linear manifold learning algorithms (PCA is used in all but the parameter analysis experiments since it is a parameter free method). Isomap [28] and Locally Linear Embeddings (LLE) [20] respectively represent global, and local, approaches to manifold learning. Finally, Local Tangent Space Alignment (LTSA) [35] represents the local/global family of techniques.

Throughout these experiments a Swiss Roll data set is used with zero added noise, apart from in Section 4.4 where noise is explicitly investigated. Section 5 contains a more detailed description of this data set.

4.1 Quality Measures

The purpose of the quality measures is to assess the stability of a manifold learning algorithm's embedding of a given dataset. Therefore the choice of appropriate quality measures is essential to gain a solid understanding of how an algorithm performs. Three different measures are used to analyse the performance of the manifold learning algorithms. Venna and Kaski rightly postulated that the retention of local properties of a dataset are often of more importance than the retention of global properties [31]. As such two of the three quality measures that are employed are concerned with the local stability of the embedding, one to measure the topological stability (Trustworthiness), and one to measure the geometric stability (Procrustes Error). The third quality measure used is concerned with how well the global distances, and therefore global structure, of the data has been maintained (Residual Variance).

Trustworthiness (TW) [31] measures the retention of neighbours between the high and low-dimensional spaces. A high value of Trustworthiness is a good measure of how well the local

neighbourhoods have been reconstructed in the low-dimensional space. Following the same terminology and methodology as in [31], the Trustworthiness of an embedding over a neighbourhood size region k is defined as

$$T_k = 1 - \frac{2}{nk(2n - 3k - 1)} \sum_{i=1}^{n} \sum_{j \in U_k(i)} (r(x_i, x_j) - k) \tag{7}$$

where U_k is the set of data samples that are in the k-neighbourhood of x_i in the low-dimensional space but are not neighbours in the high-dimensional space. $r(x_i, x_j)$ is the rank of the data sample x_j in the ordering according to the distance to x_i in the high-dimensional space. So a large value of $r(x_i, x_j)$ will occur if the point x_j enters into the neighbourhood of x_i from a long distance in the high-dimensional space. $\frac{2}{nk(2n-3k-1)}$ is a normalising term. A Trustworthiness value of 1 indicates that no local neighbourhood distortions have occurred as a result of manifold learning.

The Procrustes Error (PE) [7] measures the local distortion introduced as a result of manifold learning. Since this error metric is based on Procrustes analysis it measures the change in rotation between a neighbourhood in high-dimensional space and that same neighbourhood in the low-dimensional space. The error measure is defined as

$$R_k(\mathbf{X}, \mathbf{Y}) = \frac{1}{n} \sum_{i=1}^{n} G(x_i, y_i) \tag{8}$$

where $G(x_i, y_i)$ returns the Procrustes statistic of the neighbourhood around x_i and y_i (for computation of these values see [16, 24, 7]). A Procrustes Error value of 0 indicates no change in rotation between a given neighbourhood in the high and low-dimensional spaces. The function R_k is sensitive to scaling and as such will penalise those techniques that normalise the data or introduce local scale changes.

The Residual Variance error (RV) measures the difference in distances between all points in the high and low-dimensional space [28]. The error is measured by

$$\rho_k = 1 - R^2(\mathbf{D_X}, \mathbf{D_Y}) \tag{9}$$

where $\mathbf{D_X}$ is a square symmetric matrix containing the distances between all points in the

high-dimensional space (as measured by the geodesic distance graph formed by the k-nearest neighbours). $\mathbf{D_Y}$ is the Euclidean distance matrix between points in the low-dimensional space. R^2 is the linear correlation coefficient, taken over all entries of $\mathbf{D_X}$ and $\mathbf{D_Y}$. RV provides a good approximation of how well the manifold learning technique has preserved the distances during the manifold learning process. A value of 0 would indicate a perfect correlation between the two distance matrices.

4.1.1 Adjusting for Affine Transforms

An affine transform of a low-dimensional embedding yields a low-dimensional embedding in the same space, therefore, the above quality measures need to be adjusted for affine transformations. It is conceivable that adjusting for affine transformations produces a better low-dimensional embedding and so all of the above measures are adjusted for affine transforms by computing the minimum of the function:

$$g(\mathbf{X}) = \min_{\mathbf{A},\mathbf{x}} f(\mathbf{AY} + \mathbf{ce}^T; \mathbf{X}) \tag{10}$$

where $f(\cdot)$ is either $1 - T_k$, R_k, or ρ_k. All of the quality measures reported in the following section, except for noise in Section 4.4, are found using the minimisation of Equation 10.

4.2 Parameter Selection

The neighbourhood size parameter, k, has various uses depending on which manifold learning technique is being used. For Isomap the neighbourhood size parameter determines the number of nearest neighbours to connect when forming the neighbourhood graph. For LLE the neighbourhood size parameter determines the number of points to include in the reconstruction of a sample from its neighbours. For LTSA and PLML the neighbourhood size parameter determines the number of samples contained within the local tangent spaces or models. With this in mind it is expected that with a small k value, Isomap will fail to recover a faithful embedding as it will produce a disconnected graph, LLE will not recover enough local information to reconstruct the data-points properly, whereas the local models produced by LTSA will not contain enough local information to gain a proper approximation of the structure of the manifold. Similarly, if the

value of k is too large then Isomap's neighbourhood graph will short-circuit leading to incorrect geodesics. LLE, LTSA and PLML will over approximate the linearity of the local models and so produce incorrect mappings because they attempt to enforce linear constraints on parts of the manifold which are in fact non-linear.

Fig. 4 shows the results as a function of the neighbourhood size k with the number of data points $n = 2000$. Both PLML and Isomap outperform the other techniques with PLML obtaining the best results in all but the Residual Variance case. This is not surprising since Isomap seeks to minimise the residual variance cost function and so will out-perform other techniques when considering this metric.

4.3 Data Density

The ability of manifold learning algorithms to deal with sparse data is one of the open problems in manifold learning. At a certain level of sparsity there may be insufficient information contained within the data to reconstruct the manifold. However, algorithms should still be able to extract some meaningful structure from what little data is available. As the density of the data decreases, and so the sparsity of the data increases, algorithms should be able to gracefully deteriorate with the sparseness of the data.

Four different dataset sizes are used to represent the change of data from sparse to dense. The dataset sizes are $n = 250, 500, 1000$ and 2000 data points. All datasets are sampled from the Swiss Roll.

Table 1 shows the Trustworthiness, Procrustes Error and Residual Variance values for each dataset density value. Each of the values in Table 1 were obtained by running each manifold learning algorithm multiple times for each density value. A different neighbourhood size parameter was selected for each run in the range $k_{\text{mani}} = [5, 9, 18, 36]$. For each of these parameter values each quality measure was recorded over a range of neighbourhood size values $k_{\text{qm}} = [4, 8, 16, 32]$. The average value over different manifold learning parameter and quality measure parameter was recorded as well as the standard deviations across all values.

The general trend in results is that as the density increases the algorithms become more stable as they are able to better recover the underlying manifold. The results on different densities show

that all algorithms struggle when the data density is low ($n < 1000$) particularly when considering Residual Variance. This is unsurprising as there is often insufficient local and global information contained in the sparse data to approximate the shape of the manifold correctly. With $n \geq 1000$ PLML becomes the top performer as it is able to recover both the local and global properties of the manifold. Even though Isomap just out performs PLML when it comes to RV at high densities, PLML manages to out perform Isomap when observing local stability.

By observing the change of values in Table 1 it can be seen that the performance of PLML improves as the density increases. Similarly, as the density decreases then the performance of PLML will also decrease, but not to the extent to make it unusable. Globally, at low density, PLML is unable to recover the true manifold structure. This is not however specific only to PLML. All other manifold learning algorithms fail to uncover the global structure at low-density. As such, as long as the data is well sampled, PLML will outperform the other manifold learning algorithms.

4.4 Noisy Data

Many real world datasets are inherently noisy and so it is important for algorithms that handle such data to cope with noise in a well defined manner. Within the field of manifold learning, noise can often lead to distorted or even completely unusable embeddings. As such the ability of manifold learning algorithms to deal with noise is an important topic. The noise model used throughout these experiments is that of Gaussian noise with a mean of 0 and standard deviation in the range of $\sigma = [0, 1]$. Although with $\sigma = 1$ the Swiss Roll does not completely degrade it is still difficult for many manifold learning algorithms to pick up the true structure of the manifold with this amount of noise. Throughout the discussion $n = 2000$ is used so as to factor out the effect of density and concentrate solely on the effect of noise.

Fig. 5 shows the effect of noise on algorithm performance on the Swiss Roll dataset with results obtained using PCA added for comparison. The values for each error measure were obtained by averaging both the error measure and manifold learning parameter values. For each error measure PLML degrades with noise; as the noise increases the performance of PLML decreases. With the exception perhaps of PCA, PLML is the only algorithm that shows a well

defined and predictable decrease in performance. The other algorithms are more unpredictable in the presence of noise. For example, consider the RMSE value for LLE (Fig. 5 (c)). At $\sigma = 0$, LLE records an average equal to $\text{RMSE}_{\text{LLE}} = 0.506$, at $\sigma = 0.25$ this value increases to $\text{RMSE}_{\text{LLE}} = 0.71$ indicating a decrease in the global quality of the embedding. However as the noise increases the value of σ drops again indicating that LLE produces a better global approximation.

There is one key reason why PLML's performance drops in the presence of noise. The noise affects the clustering step and so in turn leads to not only incorrect models being formed but also an incorrect MST being produced. If the clustering step fails to produce accurate local models of the data then it is unlikely for the PLML algorithm to recover a true embedding of the manifold. Fig. 6 shows an MST created using the PLML algorithm with $n = 2000$ and $\sigma = 1$. It is clear to see that short-circuits have been introduced into the MST that cut across the manifold. These short-circuits will introduce distortions and overlaps into the embedding as the merging step will merge two models that are not topological neighbours. This is why the Trustworthiness decreases: overlaps cause points to jump between non-connected neighbours. Similarly the RMSE increases because short-circuits distort the true distances between points and models. The Procrustes Error increases but not to such an extend as is shown in Trustworthiness and RMSE. This is due in part to the models retaining their local geometric properties even if the MST is not correctly formed.

Since PLML performs well at mid to low noise (e.g. $\sigma < 0.5$) it could be worthwhile applying a noise reduction algorithm (e.g [10, 18]) prior to using PLML so as to attempt to factor out any of the effects that noise may have on the performance.

4.5 Starting Model

A key question raised by the Piecewise-Linear Manifold learning algorithm is what starting model to use in the MST traversal. The assumption is that a random model can be chosen to initiate the traversal and choosing another starting model will not change the embedding result. To address this 15 different starting points are chosen from the Swiss Roll each at a different position either around the edge or along the centre of the manifold. The starting points are labelled as shown

in Fig. 7 according to their position relative to the low-dimensional embedding. So the top left point is called TL, the middle centre point is called MC, the Bottom Right is called BR, etc. The Trustworthiness, Procrustes Error and Residual Variance are then measured for each of these embeddings. To cancel out the effect of noise, density and neighbourhood size parameter a Swiss Roll with $n = 2000$ was used and PLML's k value is set to $k = 9$ (this value falls within the optimal range of parameter values discussed above). The results are summarised in Table 2.

The bottom row of Table 2 shows the standard deviation across all starting models. A low value indicates that the starting model has no effect on this quality measure. With this in mind it can be seen that the starting model has no effect on Trustworthiness as the average standard deviation is 0.00. This is not the case for the Procrustes Error and RV. However, these standard deviations can be explained. First, the standard deviation for RV simply reflects the standard deviation within the measurement of RV for each starting model. The value of 0.07 is constant across all models and so does not reflect any change between the RV when starting at different models but rather the standard deviation from within each embedding. For the Procrustes Error the situation is similar, however there are some slight outliers, most notably BCL and BC. However for each of these starting positions the standard deviation is high within the embedding measurement suggesting that there are outlier measurements rather than a drastic change of Procrustes Error for that specific starting model.

These results show that, although there is slight deviation in the results when choosing different starting models, these deviations do not generate drastically different embeddings. As such the starting model does not have enough effect on the resulting embedding to warrant the search for an ideal starting point. Rather, a starting model can be selected at random with little change to the quality of the embedding.

5 Results

The performance of the Piecewise-Linear Manifold Learning algorithm is tested on both artificial and image based data. In Section 5.1, Piecewise-Linear Manifold Learning is compared against other leading algorithms on five benchmark synthetic data sets. Section 5.2 presents the results of running the Piecewise-Linear Manifold Learning algorithm on well known image datasets.

5.1 Artificial Datasets

Piecewise-Linear Manifold Learning (PLML) is compared against five state of the art manifold learning techniques. Isomap [28] and LLE [20] are used to represent the global, and local, approaches to manifold learning. Since PLML is a local/global approach, three leading local/global algorithms are used. LTSA [35], Manifold Charting [3] and LLC [19]. For Isomap, LLE, LTSA and PLML, the neighbourhood size parameter was set at $k = 10$. For Manifold Charting and LLC, 60 factor analysers were used and the neighbourhood size parameter for LLC was set at $k = 10$.

To visually compare the performance of these algorithms, five benchmark datasets are used: (1) the Swiss Roll, (2) Swiss Roll with a hole, (3) 2-dimensional Gaussian, (4) non-uniformly sampled Fishbowl and (5) the Helix dataset. The two Swiss Roll datasets represent a 2-dimensional plane wrapped into 3-dimensional space. As such, this dataset has become the benchmark dataset in the manifold learning literature as it is an easy to understand, yet non-linear, dataset. The Swiss Roll with a hole dataset is the same as the normal Swiss Roll dataset but with a large hole punched out of the middle. The 2-dimensional Gaussian dataset contains points drawn from a uniformly sampled Gaussian distribution in 2-dimensions. The third dimension is then taken as the relative distance of the point to the mean of the distribution such that points lying closer to the mean have a larger 'height' value than those lying on the outskirts. The non-uniform Fishbowl dataset contains points sampled from the plane $z = 0$ which are then projected onto the unit sphere. As such, the dataset resembles a fishbowl (i.e. a sphere with the top chopped off). The final dataset is the Helix dataset which is a 1-dimensional curve twisted in 3-dimensional space. All the datasets are embedded using the above algorithms into 2-dimensional space, with the exception of the Helix dataset which is embedded into 1-dimension.

Fig. 8 shows the resulting embeddings of the different manifold learning algorithms on the five datasets. Each row shows the embeddings produced by a manifold learning algorithm on the different data sets (columns). For the Swiss Roll and Swiss Roll with a Hole datasets (columns one and two), both LTSA and PLML are able to recover the best embeddings. They are able to 'un-roll' the dataset whilst retaining both the local and global properties of the manifold. LTSA does however produce a normalised embedding, which can be problematic if manifold learning

is being used as a pre-processor to some machine learning tasks [8]. Isomap is able to recover the global shape of the manifold, however there are local distortions, and for the Swiss Roll with a Hole dataset the hole is stretched as the inter-point geodesic distances across the hole are not equal to the inter-point Euclidean distances [13]. LLE, Charting and LLC all suffer from global distortions which are particularly evident when considering the embeddings produced by Charting and LLC on the Swiss Roll with a Hole dataset.

The Fishbowl dataset and the Helix dataset present two interesting problems for manifold learning. Both contain essential loops [14] which makes them difficult manifolds to embed. As well as this, the non-uniform sampling of the Fishbowl dataset can be problematic for spectral techniques such as Isomap. For both of these datasets PLML introduces a cut into the manifold. For the Helix dataset this is a desirable property, as it enables the data to be embedded into 1-dimension without any overlaps. The final column shows this as PLML is the only algorithm to be able to successfully embed the data into 1-dimension whilst retaining the labelling information. However, for the Fishbowl it can be seen that this cutting causes problems for the neighbourhood information in the low-dimensional space. The points on the bottom of the Fishbowl are embedded far away in the low-dimensional space, and the points on the rim of the Fishbowl do not form a perfect circle and instead are cut. This is an important property of the PLML algorithm to bear in mind, if the manifold does contain essential loops then it will be cut so as to embed it into the low-dimensional space. This cut arises from the use of the MST as the topological skeleton of the dataset. By definition, an MST cannot be cyclic and so it stands that for manifolds that have cycles (such as the Fishbowl and Helix datasets), the MST will at some point cut the manifold.

5.2 Image Data

Learning the low-dimensional manifold of image data presents an interesting real-world problem for manifold learning algorithms. Testing manifold learning algorithms on image data has become one of the corner stones of this particular research field (e.g. [28, 20, 32, 34]). When considering image data for manifold learning the raw pixel information is used as the input features. Each image represents a single point in high-dimensional space, so a set of 200 images of size 32×32

pixels corresponds to a set of 200 points in $\mathbb{R}^{1024}$. Three image datasets are investigated in this section. The performance of PLML on the Frey faces dataset (Section 5.2.1) is compared against existing manifold learning algorithms so as to align its performance with existing methods. The two remaining datasets are ISO faces (Section 5.2.2) and COIL-20 objects (Section 5.2.3).

5.2.1 Frey Faces

The Frey faces dataset [20] consists of a sequence of images showing a face undergoing different facial expressions. There are 1965 images each of size 20×28 pixels, meaning the data lies in $\mathbb{R}^{560}$. The only change in the data between frames is the facial expression with some left-right and up-down pose variation. The lighting is fixed. However, the amount of time spent on each facial expression varies. So for example, the neutral facial expression is over represented, whereas the tongue expression is under represented.

Fig. 9 shows the quantised embedding of the Frey faces dataset found using PLML along with comparison embeddings found using Isomap, LLE, LLC, LTSA, and Manifold Charting. All embeddings were obtained using the neighbourhood size value that produced the best possible embedding for the given manifold learning algorithm. As well as this, the Residual Variance (RV) is recorded for each algorithm to display how well the methods work at embedding the data into its intrinsic dimension. In this case, the intrinsic dimension was estimated as $q = 7$ using the Maximum Likelihood Estimation technique [15]. The quantisation process simplifies the visualisation of the embedding. Since there are too many points to display one image per point the embedding space is split into a grid and the median face of all the points contained within each grid cell is displayed. This means that an overview of the structure of the manifold can be observed without the need for all data points. Initially it is difficult to extract any structure or information from the embedding as it does not display a perfect interpolation of expressions across the embedding. However, following further examination it becomes clear that the different facial expressions have been segmented into different regions of the manifold. Due to the non-optimal target dimensionality there are some discontinuities in the mapping where some facial expressions 'jump' to different regions but generally different facial expressions reside in different areas of the manifold.

Isomap, LLE, and LTSA extract a manifold with two distinct sections relating generally to

facial expression, with happy faces occupying one arm of the embedding and neutral or sad faces occupying the other arm. LLC, Charting, and PLML obtain embeddings where different areas of the embedding contain different facial expressions. Both LLC and PLML have the neutral facial expression contained at the centre of the embedding with the more 'expressive' facial expressions radiating out from this central point.

5.2.2 ISO-faces

The ISOFaces dataset [28] is a sequence of 698 images of size 64×64 pixels showing a 3-dimensional computer rendered head under different pose and lighting conditions. The pose varies from left to right and up to down; the lighting conditions vary according to the point light's position from left to right.

Fig. 10 shows the 3-dimensional embedding of the ISO-faces dataset using PLML with $k = 12$. The low-dimensional embedding captures the change in pose as well as the change in lighting conditions. Two paths are traced through the manifold (as shown in Fig. 10) to display the change in the pose and lighting. The path moving along the horizontal axis displays the change in pose from 'looking right' to 'looking left' while the path traced perpendicular to this shows how the lighting conditions change. This second path shows that the lighting conditions change while the pose remains the same.

5.2.3 COIL-20 Object

The COIL-20 [17] object data consists of a set of 72 images of a 3-dimensional object rotating around 1 axis. Each image is 32×32 pixels in size and as such this dataset consists of a sparse problem as there are considerably more dimensions than samples (72 samples in 1024-dimensional space). Due to the fact that the object only moves around 1-axis and all other variables (e.g. object size, lighting, etc) are fixed then this data is expected to lie on a 2-dimensional manifold. This manifold is parameterised by the degree of rotation and so equates to a circular manifold embedded within the 1024-dimensional space. This estimation of intrinsic dimensionality is backed up by the average intrinsic dimensionality estimate of 2-dimensions.

The PLML based MST of the COIL-20 object dataset is shown in Fig. 11. What is immediately noticeable is that the MST is intrinsically 1-dimensional. It has no branches or sub-graphs

and simply consists of a linked list of models. The MST also successfully captures the rotation of the object with the models being connected in the correct rotational order. A link could be made between the last model in the MST and the first model to produce a continuous cycle, however this link does not exist. Due to the circular nature of the manifold, PLML will be unable to embed the data successfully into 2-dimensions as indicated by the embedding shown in Fig. 12. This embedding is not quantised as with the Frey faces and ISOFaces dataset; rather every low-dimensional image is shown. The parameter value of $k = 4$ was chosen. The embedding in Fig. 12 does exhibit some structure, however, it is difficult to draw any conclusions from such an embedding as the structure is not immediately obvious.

The COIL-20 dataset can however be viewed as a 1-dimensional manifold and therefore embedded into 1-dimensional space. The quantised result of embedding the COIL-20 dataset into 1-dimensional space using PLML is shown in Fig. 13. By embedding the data into 1-dimensional space PLML has managed to capture the structure of the data and clearly shows that the embedding is parameterised by the degree of rotation of the object. As such the structure that was not obvious in the 2-dimensional embedding is identified in this 1-dimensional representation.

6 Conclusions

In this paper a heuristic approach to manifold learning has been presented: Piecewise-Linear Manifold Learning (PLML). One of the key strengths of the PLML algorithm is its simplicity and intuitiveness. With most existing manifold learning algorithms it is very difficult to trace the execution of the algorithm as they are often concerned with the minimisation of a cost function obtained through the solution of a sparse eigenproblem. PLML on the other hand builds the global low-dimensional embedding in an intuitive manner by merging local models in a topologically pre-defined manner according to the Minimum Spanning Tree of the models. This makes it easier to trace the execution of the PLML algorithm. PLML also does not require the solution to a large dynamic system which simplifies its computational complexity and runtime. If appropriate algorithms are used for the building of the Minimum Spanning Tree (e.g. [4]) and also for the matrix multiplication (e.g. [26, 5]) then PLML can be used for large datasets which

normal manifold learning algorithms with runtime of $O(n^3)$ would not be able to handle.

It has been shown that PLML is able to produce competitive results compared with existing state-of-the-art manifold learning algorithms on both artificial and image datasets. PLML differs from existing global alignment of local models techniques, such as LTSA [35], Manifold Charting [3] and Locally Linear Coordination [19], as it uses a hard partitioning of the data to model the manifold at a local scale. It also does not rely on any form of optimisation to perform alignment of these local models. As such, it will not get caught in any local minima and does not require the solution to a large scale eigenproblem.

Future areas of research will focus on the application of PLML to different imaging applications with a particular focus on how the local modelling of the manifold can be used to incorporate PLML into a larger classification system. As well as this, extensions to PLML could be made by investigating different approaches to model construction and alignment, such as replacing the MST traversal with other graph search algorithms.

Acknowledgements

This work was part funded by the Aberystwyth APRS Award and the NISCHR Biomedical Research Unit for Advanced Medical Imaging and Visualisation. Thanks also goes to Dr Frédéric Labrosse, Dr Richard Jensen and Dr Neil Mac Parthaláin for their valuable discussions in the preparation of this work.

References

[1] M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14: Proceedings of the 2002 Conference (NIPS)*, pages 585–591, 2002.

[2] K. P. Bennett, P. S. Bradley, and A. Demiriz. Constrained k-means clustering. Technical Report MSR-TR-2000-65, Microsoft Research, May 2000.

[3] M. Brand. Charting a manifold. In *Advances in Neural Information Processing Systems 15: Proceedings of the 2003 Conference (NIPS)*, pages 961–968, 2003.

[4] K. W. Chon, Y. Han, and W. Tak. Concurrent threads and optimal parallel minimum spanning trees algorithms. *Journal of the Association for Computing Machinery*, 48(2):297–323, 2001.

[5] D. Coppersmith and S. Winograd. Matrix multiplication via arithmetic progressions. *Journal of Symbolic Computation*, 9(3):251–280, 1990.

[6] T. F. Cox and M. A. A. Cox. *Multidimensional Scaling*. Chapman and Hall, 2001.

[7] Y. Goldberg and Y. Ritov. Local Procrustes for manifold embedding: a measure of embedding quality and embedding algorithms. *Machine Learning*, 77(1):1–25, 2009.

[8] Y. Goldberg, A. Zakai, D. Kushnir, and Ya'acov Ritov. Manifold learning: The price of normalization. *Journal of Machine Learning Research*, 9(1909-1939), 2008.

[9] J. C. Gower. Generalized Procrustes Analysis. *Psychometrika*, 40(1):33–51, 1975.

[10] M. Hein and M. Maier. Manifold denoising. In *Advances in Neural Information Processing Systems 18: Proceedings of the 2006 Conference (NIPS)*, 2006.

[11] H. Hotelling. Analysis of a complex of statistical variables into principal components. *Journal of Educational Psychology*, 24:417–441, 1933.

[12] S. Lafon and A. B. Lee. Diffusion maps and coarse-graining: A unified framework for dimensionality reduction, graph partitioning, and data set parameterization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(9):1393–1403, 2006.

[13] J. A. Lee and M. Verleysen. *Nonlinear Dimensionality Reduction*. Springer, 2007.

[14] John Aldo Lee and Michel Verleysen. Nonlinear dimensionality reduction of data manifolds with essential loops. *Neurocomputing*, 67:29–53, 2005.

[15] E. Levina and P. J. Bickel. Maximum likelihood estimation of intrinsic dimension. In *Advances in Neural Information Processing Systems 16: Proceedings of the 2004 Conference (NIPS)*, 2004.

[16] K. Mardia, J. Kent, and J. Bibby. *Multivariate Analysis*. New York: Academic, 1979.

[17] S. A. Nene, S. K. Nayar, and H. Murase. Columbia Object Image Library (COIL-20). Technical report, Technical Report CUCS-005-96., 1996.

[18] J. Park, Z. Zhang, and H. Zha. Local smoothing for manifold learning. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[19] S. Roweis, L. K. Saul, and G. E. Hinton. Global coordination of local linear models. In *Advances in Neural Information Processing Systems 14: Proceedings of the 2002 Conference (NIPS)*. MIT Press, 2002.

[20] S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by Locally Linear Embedding. *Science*, 290:2323–2326, 2000.

[21] B. Scholkopf, E. Smola, L. Bottou, C. Burges, H. Bultho, K. Gegenfurtner, and P. Ha Ner. Nonlinear component analysis as a kernel eigenvalue problem. *Neural Computation*, 10:1299–1319, 1998.

[22] P. H. Schonemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31:1–10, 1966.

[23] H. S. Seung and D. D. Lee. The manifold ways of perception. *Science*, 290(5500):2268–2269, 2000.

[24] R. Sibson. Studies in the robustness of multidimensional-scaling: Procrustes statistics. *Journal of the Royal Statistical Society*, 40(2):234–238, 1978.

[25] H. Strange and R. Zwiggelaar. Classification performance related to intrinsic dimensionality in mammographic image analysis. In *Proceedings of the Thirteenth Annual Conference on Medical Image Understanding and Analysis*, pages 219–223, 2009.

[26] V. Strassen. Gaussian elimination is not optimal. *Numerical Mathematics*, 13:354–546, 1969.

[27] Y. W. Teh and S. Roweis. Automatic Alignment of Hidden Representations. In *Advances in Neural Information Processing Systems 15: Proceedings of the 2003 Conference (NIPS)*, pages 841–848, 2003.

[28] J. B. Tenenbaum, V. de Silva, and J. C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290:2319–2322, 2000.

[29] G. Valiente. *Algorithms on Trees and Graphs*. Springer-Verlag, Berlin Heidelberg, 2002.

[30] L. van der Maaten, E. Postma, and J. van den Herik. Dimensionality reduction: A comparitive review. Technical Report TiCC-TR 2009-005, Tilburg University, 2009. Unpublished.

[31] J. Venna and S. Kaski. Local Multidimensional Scaling. *Neural Networks*, 19:889–899, 2006.

[32] J. Verbeek. Learning non-linear image manifolds by global alignment of local linear models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(8):1236–1250, 2006.

[33] K. Q. Weinberger and L. K. Saul. An introduction to nonlinear dimensionality reduction by Maximum Variance Unfolding. In *Proceedings of the Twenty-First AAAI Conference on Artificial Intelligence*, pages 1683–1686, 2006.

[34] K. Q. Weinberger and L. K. Saul. Unsupervised learning of image manifolds by semidefinite programming. *International Journal of Computer Vision*, 70:77–90, 2006.

[35] Z. Zhang and H. Zha. Principal manifolds and nonlinear dimension reduction via local tangent space alignment. *SIAM Journal on Scientific Computing*, 26(1):313–338, 2004.

Tables

		TW	PE $^{(\times 10^{-4})}$	RV
$n = 250$	ISO	0.901 ± 0.01	1.073 ± 0.54	0.599 ± 0.22
	LLE	0.879 ± 0.03	1.102 ± 0.61	0.663 ± 0.01
	LTSA	0.874 ± 0.05	1.156 ± 0.36	0.695 ± 0.12
	PLML	0.826 ± 0.10	1.126 ± 0.30	0.741 ± 0.05
$n = 500$	ISO	0.902 ± 0.06	1.112 ± 0.63	0.448 ± 0.35
	LLE	0.863 ± 0.01	1.035 ± 0.93	0.625 ± 0.13
	LTSA	0.845 ± 0.06	0.842 ± 0.46	0.773 ± 0.10
	PLML	0.932 ± 0.08	0.784 ± 0.33	0.656 ± 0.24
$n = 1000$	ISO	0.944 ± 0.05	1.422 ± 0.31	0.188 ± 0.20
	LLE	0.891 ± 0.01	0.973 ± 0.81	0.614 ± 0.13
	LTSA	0.846 ± 0.12	0.732 ± 0.41	0.663 ± 0.30
	PLML	0.980 ± 0.04	0.288 ± 0.36	0.270 ± 0.38
$n = 2000$	ISO	0.985 ± 0.03	0.806 ± 0.82	0.030 ± 0.05
	LLE	0.890 ± 0.01	1.207 ± 0.65	0.513 ± 0.25
	LTSA	0.790 ± 0.11	0.663 ± 0.44	0.782 ± 0.17
	PLML	0.990 ± 0.02	0.064 ± 0.07	0.245 ± 0.50

Table 1: Average results of the quality measures over different density scales. The results were averaged over the manifold learning neighbourhood size range $k_{\text{mani}} = [3, 5, 9, 18, 36]$ and the quality measure neighbourhood size range $k_{\text{qm}} = [4, 8, 16, 32]$.

	TW	PE $^{(\times 10^{-6})}$	RV
TL	0.999 ± 0.00	2.967 ± 0.96	0.100 ± 0.07
ML	0.999 ± 0.00	2.437 ± 0.67	0.099 ± 0.07
BL	0.999 ± 0.00	2.855 ± 0.84	0.100 ± 0.07
TCL	1.000 ± 0.00	2.889 ± 0.91	0.100 ± 0.07
MCL	0.999 ± 0.00	2.525 ± 0.83	0.100 ± 0.07
BCL	0.999 ± 0.00	8.198 ± 3.35	0.101 ± 0.07
TC	0.999 ± 0.00	2.496 ± 0.63	0.100 ± 0.07
MC	1.000 ± 0.00	2.037 ± 0.52	0.100 ± 0.07
BC	0.999 ± 0.00	4.667 ± 5.40	0.100 ± 0.07
TCR	0.999 ± 0.00	3.349 ± 0.74	0.101 ± 0.07
MCR	0.999 ± 0.00	3.717 ± 1.37	0.100 ± 0.07
BCR	0.999 ± 0.00	2.274 ± 0.39	0.100 ± 0.07
TR	0.999 ± 0.00	2.823 ± 0.83	0.101 ± 0.07
MR	1.000 ± 0.00	2.351 ± 0.50	0.100 ± 0.07
BR	0.999 ± 0.00	3.205 ± 0.97	0.100 ± 0.07
Δ	± 0.00	$\pm 2.31 \times 10^{-6}$	± 0.07

Table 2: Average results of Trustworthiness, Procrustes Error and Residual Variance over different starting positions. The values were obtained on a Swiss Roll with $n = 2000$ and zero added noise with PLML $k = 9$.

Figure Captions

Fig. 1: Process diagram showing the basic workings of the PLML diagram. After the data has been clustered, an MST is built over all of the clusters and a traversal algorithm is used to visit all of the nodes within the MST. If the node has not been visited then that cluster is project onto the global model (forward-projection), otherwise the global model is aligned to the cluster (backward-projection). Once all of the nodes have been visited PCA is applied to the global model to produce the low-dimensional embedding.

Fig. 2: A simple example of how the iterative merging method of PLML works in 2-dimensions. As the graph traversal takes place, each previously visited model is projected onto the basis vectors of the current model so that a global model of the data is iteratively formed.

Fig. 3: The backward projection step where the previously visited models (shown as squares) need to be aligned to the current model (shown as crosses), however, as seen in (a), the two hyperplanes are too far apart and attempting to align them using forward projection would cause distortions. Rather, the current model's representation in the global model is aligned to its true representation (b).

Fig. 4: Average results of (a) Trustworthiness, (b) Procrustes Error, and (c) Residual Variance as a function of the neighbourhood size k. The results were averaged over error metric parameter value and tested on a Swiss Roll with a density of $n = 2000$ and zero added noise.

Fig. 5: The effect of noise on a densely sampled Swiss Roll ($n = 2000$) when measuring average Trustworthiness, Procrustes Error and RV. The values were obtained at each noise level over an average of error measure parameters and manifold learning neighbourhood size parameters.

Fig. 6: The effect of noise on the creation of the MST within the PLML algorithm.

Fig. 7: The different starting points used to test whether the starting model affects the

low-dimensional embedding. The points are shown as their respective positions on the low-dimensional embedding.

Fig. 8: The results of various manifold learning algorithms on five different data sets. From left to right they are, the Swiss Roll, Swiss Roll with a hole taken out, 2-dimensional Gaussian, non-uniformly sampled fishbowl and the Helix. For Isomap, LLE, LTSA and PLML, the neighbourhood size parameter was set at $k = 10$. For Manifold Charting and LLC, 60 factor analysers were used and the neighbourhood size parameter for LLC was set at $k = 10$.

Fig. 9: The quantised 2-dimensional embeddings of the Frey faces dataset found using (a) Isomap, (b) LLC, (c) LLE, (d) LTSA, (e) Manifold Charting, (f) PLML, along with the measured Residual Variance when embedding into the dataset's intrinsic dimensionality.

Fig. 10: The 3-dimensional embedding of the ISO-faces dataset obtained using PLML with $k = 12$. Two different paths are traced along the low-dimensional manifold showing how the manifold captures the change in pose and lighting condition.

Fig. 11: The MST of the Coil-20 Object dataset. The MST is intrinsically 1-dimensional, that is it has no branches or possible subgraphs. The ordering here shows that the MST correctly captures the rotational ordering of the object.

Fig. 12: The low-dimensional embedding of the COIL-20 dataset produced by PLML with $k = 4$. Notice that the circular structure of the data is not recovered.

Fig. 13: The quantised embedding of the COIL-20 dataset produced using PLML. Notice that the rotation of the object is captured in the 1-dimensional embedding whereas this structure is less apparent in the 2-dimensional embedding (Fig. 12).

Figures

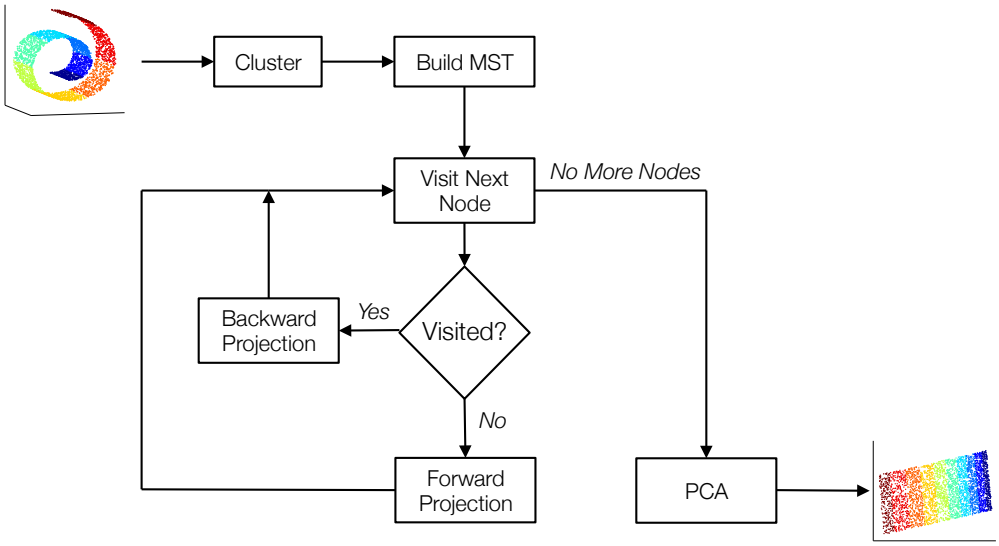


Fig. 1:

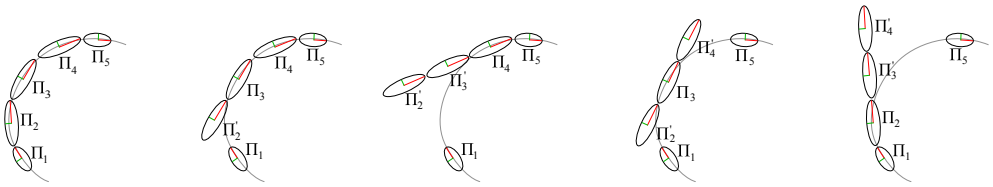
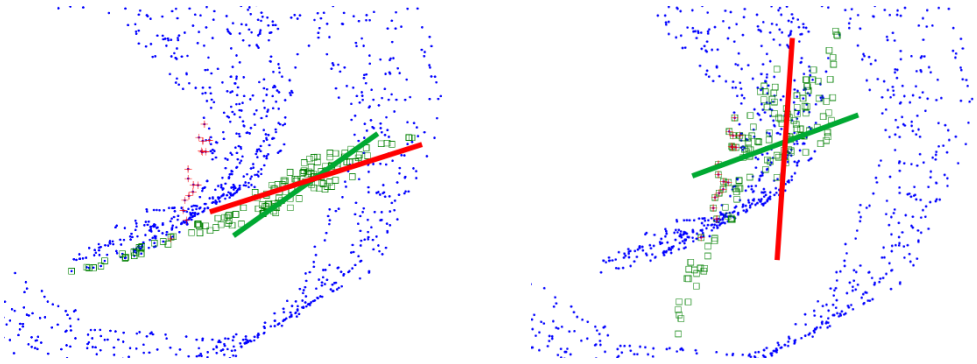
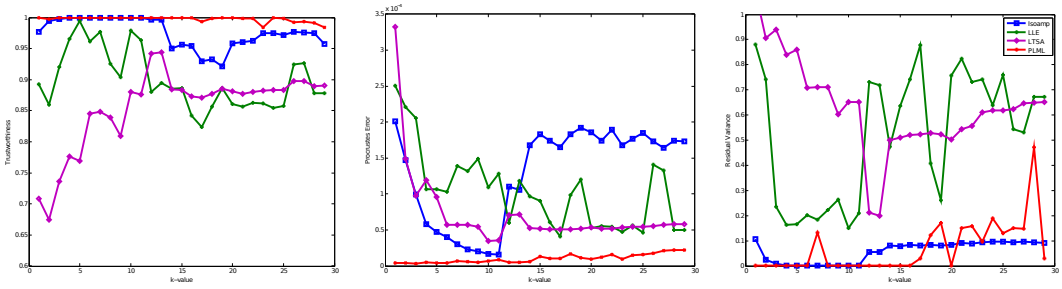

Π4
Π5
Π3
Π2
Π1
(a)
Π4
Π5
Π3
Π2'
Π1
(b)
Π4
Π5
Π3'
Π2'
Π1
(c)
Π4'
Π5
Π3
Π2'
Π1
(d)
Π4'
Π5
Π3'
Π2
Π1
(e)

Fig. 2:



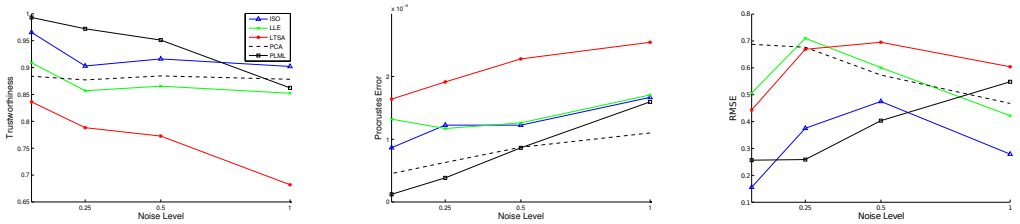
(a) (b)

Fig. 3:


Trustworthiness
k-value
Procrustes Error
k-value
Residual Variance
k-value
Isoamp
LLE
LTSA
PLML

(a) (b) (c)

Fig. 4:



(a) Trustworthiness (b) Procrustes Error (c) RV

Fig. 5:

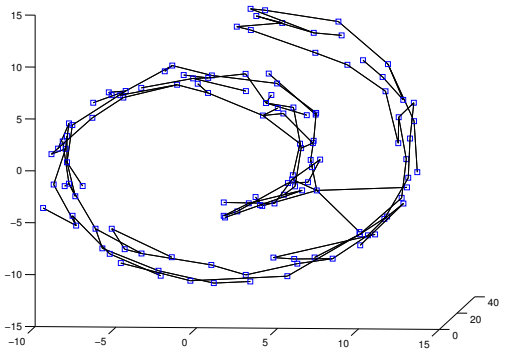


Fig. 6:

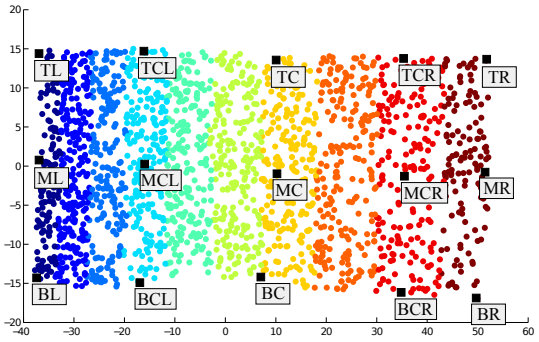

TL
TCL
TC
TCR
TR
ML
MCL
MC
MCR
MR
BL
BCL
BC
BCR
BR
20
15
10
5
0
−5
−10
−15
−20
−40
−30
−20
−10
0
10
20
30
40
50
60

Fig. 7:

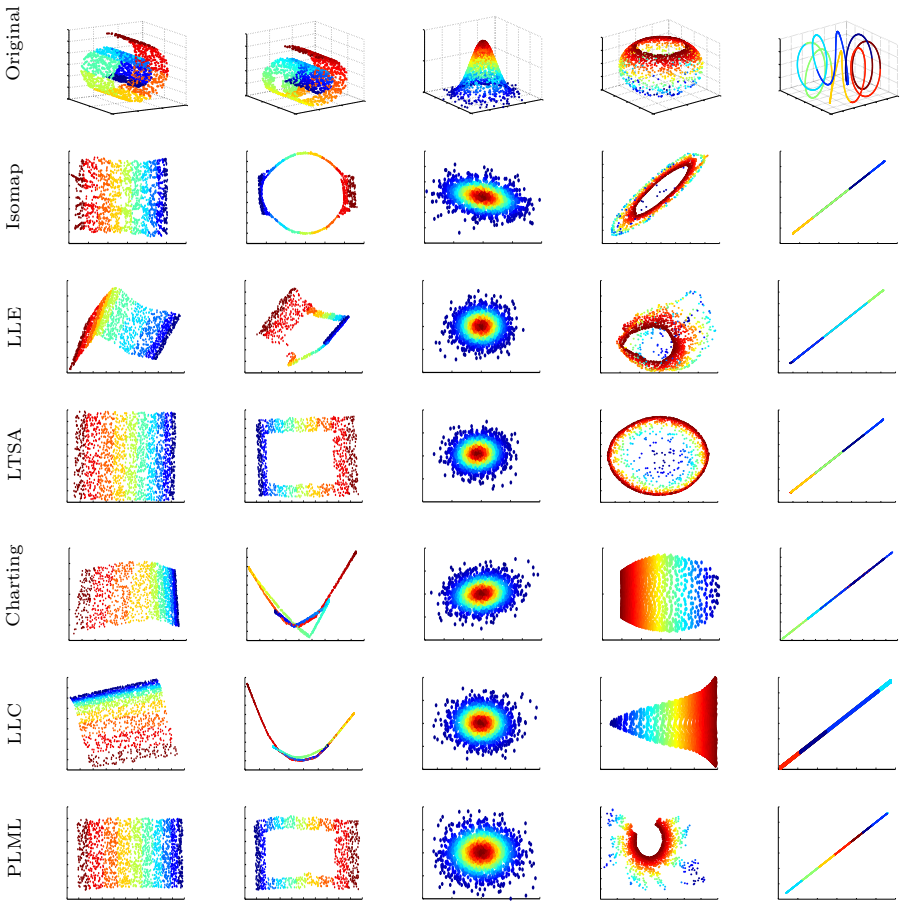
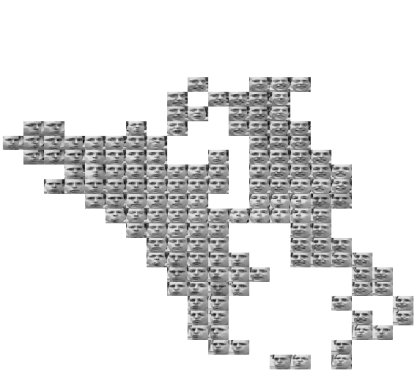
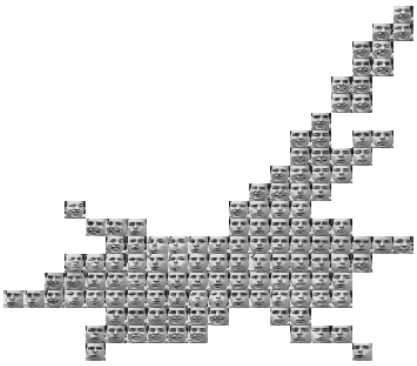


Fig. 8:

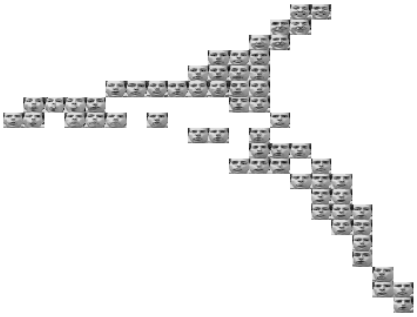


(a) RV = 0.15



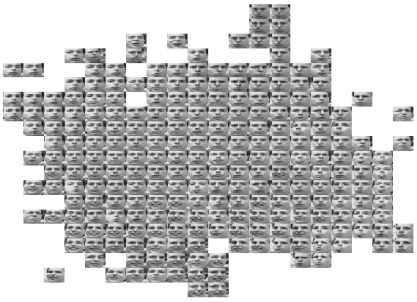
(b) RV = 0.82

(c) RV = 0.34



(d) RV = 0.48

(e) RV = 0.48



(f) RV = 0.32

Fig. 9:

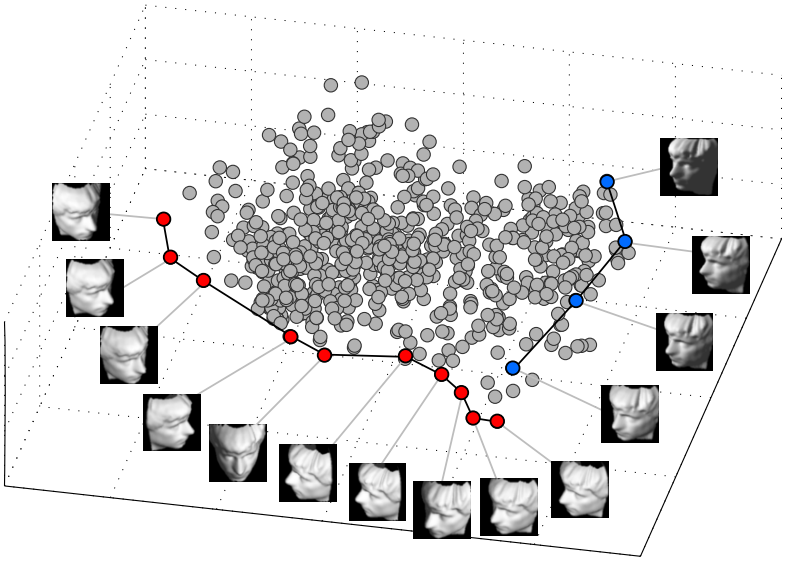


Fig. 10:

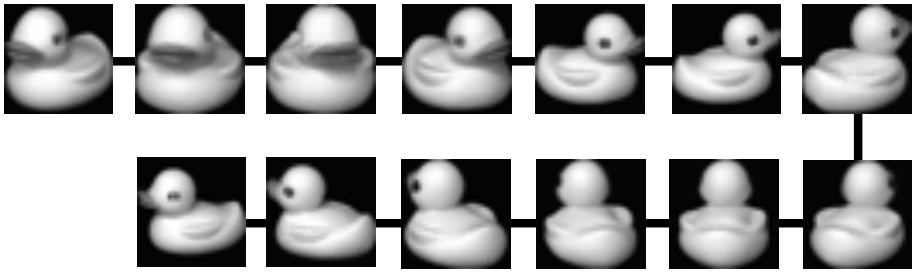


Fig. 11:

Fig. 12:

Fig. 13: